

ON THE LEXICOGRAPHIC REPRESENTATION OF NUMBERS

VINCENZO MANCA
UNIVERSITY OF VERONA - ITALY

Abstract It is proven that, contrarily to the common belief, the notion of zero is not necessary for having positional representations of numbers. Namely, for any positive integer k , a positional representation with the symbols for $1, 2, \dots, k$ is given that retains all the essential properties of the usual positional representation of base k (over symbols for $0, 1, 2, \dots, k-1$). Moreover, in this zero-free representation, a sequence of symbols identifies the number that corresponds to the order number that the sequence has in the ordering where shorter sequences precede the longer ones, and among sequences of the same length the usual lexicographic ordering of dictionaries is considered. The main properties of this lexicographic representation are proven and conversion algorithms between lexicographic and classical positional representations are given. Zero-free positional representations are relevant in the perspective of the history of mathematics, as well as, in the perspective of emergent computation models, and of unconventional representations of genomes.

Keywords

Positional representation, Zero, Lexicographic ordering

1. NUMBER POSITIONAL REPRESENTATIONS

The positional representation of numbers was an epochal discovery that is a landmark in the history of mathematics and science. It is based on a notational system of Hindu origin that, via the Arabian mathematics, was imported in Europe with the famous book *Liber Abaci* published in 1202 by Leonardo Fibonacci (Leonardo from Pisa). This number representation was the seed of a new mathematical perspective, algebraic and algorithmic, that surely contributed to the important discoveries of Renaissance mathematics, from which modern mathematics stemmed.

An important aspect of positional representation was the notion of zero (from an arabian root common to the noun *zephyr*, a gentle wind whose origin is hardly determinable). Any natural number can be univocally represented as sum of powers of any base $k > 1$ multiplied by coefficients smaller than k . From this unicity it easily follows that, given an alphabet of k symbols (standing for $0, 1, 2, \dots, k-1$), every number is univocally represented by a sequence of symbols in this alphabet, where the position of the symbol encodes a power of the basis k , and the symbol in that position encodes a multiplicative coefficient, in such a way that the

sum of these multiplied powers yields the number. When a (symbol for) zero occurs, then the corresponding power gives a null contribution to the overall sum providing the number. Classical texts in number theory and history of mathematics usually connect intrinsically the notion of zero with the positional notation [10, 3]. For example, in [10] it is written that “The only complication which the positional notation involves lies in the necessity of introducing a *zero symbol* to express a void or missing class; for instance, 204 is different from 24”. The same “necessity” is expressed by [3], and certainly the attitude toward this conviction was enforced by the decimal notation and the related algorithms for basic operations (using alignment, carry, positional shift, and decimal point), after the seminal works of François Viète and Simon Stevin, in sixteenth century [2]. However, in [11], in the context of the historical analysis of number representation systems, some remarks clearly suggest that positional notations were present in some ancient number systems. One of them is the Greek one, especially in the sexagesimal notation used in Ptolemy’s *Almagest*, where some kind positional concepts, made possible the complex computations required in the contexts of geometrical and astronomical problems. In [7] an illuminating synthesis of the historical development of number representation is reported. In passing, we recall that, in a geographically far culture, and chronologically antecedent, Maya developed a positional number system, with base twenty, also having a well defined notion of zero [5].

In this note we show that the connection between zero and the positional notation is not necessary. Namely, we prove that the usual lexicographic representation of strings, in a given alphabet, is a positional number system, but without any symbol for zero. The essence of positional notations is the notion of cipher sequence, where ciphers have positional values corresponding to the powers of the base. The main advantage of this notational mechanism is that arithmetical operations (firstly sum and multiplication) can be calculated by **operation tables**, that is, by only knowing the values of the given operation when its arguments range over a finite set of numbers (whose cardinality is equal to the value of base). We will show that just this mechanism is behind the lexicographic representation by strings, where the number n is represented by the string in position n according to the lexicographic enumeration. In [1] the possibility of a *zeroless* positional system, and its connection with the string ordering, was already investigated. However, the analysis developed in [1] is very different from the one we present in this paper, and the main properties of a lexicographic representation are determined in a completely different way. Moreover, the representation algorithms are here defined in a more direct manner (without passing through the classical positional systems as in [1]), and the correctness proofs are formally established (in [1] they are informally motivated). Finally, here, translation functions between lexicographic and classical positional representations are formally defined. It is interesting to remark that the main idea of the present paper arose during the search for unconventional genome representation, where DNA sequences are viewed as numbers (in base four). In that context the following question was posed: *Define a function that assigns to any string, over a given alphabet, its order in the lexicographic enumeration.* The solution to this question is given by Equation (1) of the “Lexicographic

A, C, G, T, AA, AC, AG, AT, CA, CC, CG, CT, GA, GC, GG, GT, TA,
 TC, TG, TT, AAA, AAC, AAG, AAT, ACA, ACC, ACG, ACT, AGA,
 AGC, AGG, AGT, ATA, ATC, ATG, ATT, CAA, CAC, CAG, CAT.

TABLE 1. The first 40 strings over $\{A, C, G, T\}$ in the lexicographic order.

Theorem” of Section 2. From the recursive formula (1), the iterative formula of Equation (2) easily follows, which directly provides a number positional representation with the properties discussed along the paper.

2. THE LEXICOGRAPHIC NUMBER REPRESENTATION

Let $S = \{a_1, a_2, \dots, a_k\}$ a finite alphabet of symbols where we define an enumeration function ω of its symbols such that $\omega(a_i) = i$, for $1 \leq i \leq k$. We denote by S^* the set of strings, that is, of finite sequences over S . In this paper we denote by $|S|$ the cardinality of the set S , while $|\alpha|$ denotes the length of string α , moreover string concatenation is expressed by juxtaposition, or by $*$ when it avoids confusion, and $\alpha(i)$ will denote the symbol of α in position i (the first position is 1 and the last equals the length of the string).

We can define a strict order over S^* by the following conditions:

$$\begin{aligned} \alpha < \beta & \quad \text{if} \quad |\alpha| < |\beta| \\ \alpha < \beta & \quad \text{if} \quad |\alpha| = |\beta| \text{ and } \exists j : \forall i < j : \alpha(i) = \beta(i) \text{ and } \omega(\alpha(j)) < \omega(\beta(j)). \end{aligned}$$

The second condition is the usual criterion adopted in the enumeration of the items in a lexicon. The first one is added to the second, in order to obtain a linear ordering over (the infinite set of) all strings over the alphabet. For example, given the (ordered) alphabet $\{A < C < G < T\}$ (typical of genomes), then Table 1 gives the first 40 strings over this alphabet enumerated in the lexicographic order:

Theorem 1 (The lexicographic Theorem). *Let S be an (ordered) finite alphabet. If $x \in S$, and $\alpha \in S^*$, then Equation (1) inductively defines the enumeration number $\omega_S(\alpha)$ of string $x\alpha$, shortly indicated by $\omega(\alpha)$, according to the lexicographic ordering on S^* ($\omega(\lambda) = 0$, if λ is the empty string):*

$$(1) \quad \omega(x\alpha) = \omega(x)|S|^{|\alpha|} + \omega(\alpha)$$

moreover:

$$(2) \quad \omega(\alpha) = \sum_{i=1}^{|\alpha|} \omega(\alpha(i))|S|^{|\alpha|-i}$$

Proof. In fact, let us denote by $\omega_=(\alpha)$ the position of α when we lexicographically enumerate the strings of length α ($a^{|\alpha|}$ is the first string in position 1, if a is the first symbol

of S). In order to evaluate the position of $x\alpha$ in the lexicographic enumeration, we observe that:

i) $x\alpha$ follows all the strings of length $\leq |\alpha|$, which are in number:

$$\sum_{i=1}^{|\alpha|} |S|^i$$

ii) and $x\alpha$ follows the strings of length $|\alpha| + 1$ beginning with any symbol y such that $\omega(y) < \omega(x)$, which are in number:

$$[\omega(x) - 1]|S|^{|\alpha|}$$

iii) between the first string of length $|\alpha| + 1$ that begins with x and $x\alpha$ (extremes included) there are $\omega_{=}(x)$ strings. Therefore, the position of $x\alpha$ in the lexicographic enumeration is:

$$\omega(x\alpha) = \sum_{i=1}^{|\alpha|} |S|^i + [\omega(x) - 1]|S|^{|\alpha|} + \omega_{=}(x)$$

that can be rewritten as:

$$\sum_{i=1}^{|\alpha|-1} |S|^i + |S|^{|\alpha|} + [\omega(x) - 1]|S|^{|\alpha|} + \omega_{=}(x)$$

but, summing the two terms in the middle of the sum above, we get the term $\omega(x)|S|^{|\alpha|}$, while summing the other two (extremal) terms we get $\omega(x)$. This concludes the proof of Equation (1) (and of the inductive definition of ω). The proof of Equation (2) follows directly, by iteratively applying Equation (1). If k is the cardinality of S , then Equation (2) can be rewritten as follows:

$$\omega(\alpha) = \sum_{i=1}^{|\alpha|} \omega(\alpha(i))k^{|\alpha|-i} \quad \text{QED.}$$

An interesting simple consequence of the theorem above is the following corollary, due to the fact that, being the lexicographic order a total ordering over all the strings, every string represents the number of its lexicographic enumeration.

Corollary 2. *Given a number $k > 0$, any non-null natural number n is univocally representable by a linear combination of powers k^i , $c_0k^0 + c_1k^1 + \dots + c_jk^j$ where $0 < c_i \leq k$ for all $0 \leq i \leq j$.*

We remark that according to the corollary above, lexicographic number representation results a *full positional representation* because all the powers of k less than a maximum value give a positive contribution to the sum representing the number, while classical representations admit the possibility of having empty contribution (expressed by zero ciphers). The

+	A	C	G	T
A	C	G	T	AA
C	G	T	AA	AC
G	T	AA	AC	AG
T	AA	AC	AG	AT

TABLE 2. Addition table of number lexicographic representation (four symbols).

$\omega(CAT)$	=	40
$\omega(GATT)$	=	228
$CAT + GATT$	=	$GTCT$
$\omega(GTCT)$	=	$40 + 228 = 268$

TABLE 3. The additivity of the lexicographic order.

lexicographic representation is the most economical way to represent the set of the first m numbers. In other words, if the cost of representing these numbers is defined (with respect to a base k) as the sum of the lengths of strings representing them, then no representation exists that has a lower representation cost.

Table 2 defines operation $+$ for symbols A, C, G, T that is coherent with the lexicographic ordering, as illustrated by Table 3. In fact, by using Table 2 we can coherently add strings, in perfect accordance with their lexicographic order.

In general the additive table of a lexicographic system can be obtained in accordance to the following proposition that follows directly from the previous discussion. In the following, when we want to avoid confusion, the symbols of a k -lexicographic representation (in base k) are denoted by $[1], [2], \dots, [k]$ (with numbers in decimal notation inside brackets), while symbols of a classical k -positional representation (with 0) are denoted by $[0], [1], \dots, [k-1]$.

Proposition 3. *Given a lexicographic system of base k , its additive table $L_k(+)$ can be obtained from the additive table $P_k(+)$ of the classical positional table: i) by removing all the elements corresponding to sums $[j] + [0]$ or $[0] + [j]$, for $j < k$; ii) by replacing in $P_k(+)$ any sum result $[1][0]$ by $[k]$; and iii) by adding as results of the sums $[j] + [k]$ or $[k] + [j]$, for $j \leq k$, the elements $[1][j]$.*

The following algorithm provides the string that lexicographically represents the number n over an alphabet of k symbols. It provides the inverse information with respect to the theorem above. In fact, given a positive integer n , it solves the equation of unknown α :

$$\omega(\alpha) = n.$$

Algorithm 4 (Lexicographic Number Representation). *For any positive integer h , let*

$$\maxlex(k, h) = \sum_{i=1}^h k^i$$

$$\minlex(k, h) = \sum_{i=1}^h k^{i-1}.$$

The first number expresses the greatest number lexicographically representable by strings of length h , while the second number expresses the smallest number lexicographically representable by strings of length h . Then, given a positive integer n , if

$$\minlex(k, h) \leq n \leq \maxlex(k, h)$$

then it is lexicographically representable by a string of length h , and we denote this value h by $\text{lex}(k, n)$, or simply by $\text{lex}(n)$ when k is implicitly understood. In this case, n is univocally lexicographically represented by the string $\sigma_k(n)$ (over $\{[1], [2], \dots, [k]\}$), according to the following recursive procedure (here $$ denotes string concatenation and \cdot the product):*

$$\begin{aligned} \sigma_k(n) &= [n] \text{ if } n \leq k \\ \sigma_k(n) &= [m] * \sigma_k(n - m \cdot k^{\text{lex}(n)-1}) \text{ otherwise} \\ \text{where } m &= \max\{q \leq k \mid n - (q \cdot k^{\text{lex}(n)-1}) \geq \minlex(k, \text{lex}(n) - 1)\}. \end{aligned}$$

QED.

For example, let us consider 37 and 36, of course $\text{lex}(4, 36) = \text{lex}(4, 37) = 3$, but 37 is lexicographically represented (in the base 4) by:

$$37 = 2 \cdot 4^2 + 1 \cdot 4^1 + 1 \cdot 4^0$$

because $37 - 2 \cdot 4^2 = 5$ and $5 \geq \minlex(4, 2)$, while:

$$36 = 1 \cdot 4^2 + 4 \cdot 4^1 + 4 \cdot 4^0$$

because $36 - 2 \cdot 4^2 = 4$ and $4 < \minlex(4, 2)$.

Corollary 5. *Given a positive integer n , the previous algorithm correctly provides the string $\sigma_k(n)$, representing n in the lexicographic system of base k , such that:*

$$\omega(\sigma_k(n)) = n.$$

Proof. The asserted correctness easily follows, by induction, from the definition of lex , \minlex , \maxlex , and the recursive procedure defining $\sigma_k(n)$. QED.

\times	1	2	3	4	5	6	7	8	9	X
1	1	2	3	4	5	6	7	8	9	X
2	2	4	6	8	X	12	14	16	18	1X
3	3	6	9	12	15	18	21	24	27	2X
4	4	8	12	16	1X	24	28	32	36	3X
5	5	X	15	1X	25	2X	35	3X	45	4X
6	6	12	18	24	2X	36	42	48	54	5X
7	7	14	21	28	35	42	49	56	63	6X
8	8	16	24	32	3X	48	56	64	72	7X
9	9	18	27	36	45	54	63	72	81	8X
X	X	1X	2X	3X	4X	5X	6X	7X	8X	9X

TABLE 4. The multiplication table of the lexicographic representation in base ten, where the first nine symbols 1, 2, 3, 4, 5, 6, 7, 8, 9 have the same value as in the classical decimal system, while X denotes ten (zero is not present).

The correctness of a number representation ρ implies that any (binary) operation \odot on numbers, when performed on number representations, has to satisfy the *morphism condition* (in the algebraic sense), that is, ρ has to commute with the operations (on numbers and representations, for the sake of simplicity both denoted by \odot):

$$(3) \quad \rho(n) \odot \rho(m) = \rho(n \odot m).$$

Multiplication in lexicographic representation can be computed with the usual algorithm of classical positional systems, by using the multiplication Table 4. For example, the multiplication $37 \times 3X = 147X$ can be obtained with the usual multiplication algorithm of the positional representation (147X corresponds to 1480 in the usual decimal system, coherently to the correspondence of $(37, 3X)$ to the decimal pair $(37, 40)$ and to the decimal multiplication $37 \times 40 = 1480$).

Analogously to the additive table, the multiplicative table of a lexicographic system can be obtained in accordance to the following proposition.

Proposition 6. *Given a lexicographic system of base k , its multiplicative table $L_k(\times)$ can be obtained from the multiplicative table $P_k(\times)$ of the classical positional table i) by removing all the elements corresponding to multiplications $[j] \times [0]$ or $[0] \times [j]$, for $j < k$; ii) by replacing in $P_k(\times)$ any multiplication result $[1][0]$ by $[k]$, and any multiplication result $[j][0]$ by $[j - 1][k]$; iii) by adding as results of the multiplications $[j] \times [k]$ or $[k] \times [j]$, for $j \leq k$, the elements $[j - 1][k]$.*

Corollary 7. *Given a positive integer n , in the lexicographic system of base k (symbols $[1], [2], \dots, [k]$) the result of multiplication $\sigma_k(n) \times [k]$ is represented by $\sigma_k(n - 1)[k]$ ($\sigma_k(0)$ is the empty string).*

$$\begin{array}{r}
423 \times \\
8X = \\
\hline
422X \\
3384 \\
\hline
37X6X
\end{array}$$

TABLE 5. A multiplication in the lexicographic representation of base ten. In the corresponding zero positional system the same multiplication becomes 423×90 , which usually is performed by adding a final 0 to $423 \times 9 = 3807$. Of course 38070 transforms, according to Algorithm 8, into $37X6X$ (and *viceversa*). It seems that the usual multiplication is much simpler than the lexicographic one. However, also in the lexicographic system it is possible to get the result in a shorter way. In fact, the multiplication $423 \times X$ can be obtained, according to Corollary 7, by appending X to the predecessor of 423, getting $422X$, then $423 \times 8 = 3384$, therefore, shifting and summing the two partial results, the final result is obtained.

3. LEXICOGRAPHIC AND ZEROPOSITIONAL REPRESENTATIONS

Rules can be established that transform a lexicographic representation, in a given base, into a classical (with zero) positional representation in the same base and *vice versa*. Let us denote by $\delta_k(n)$ the k -positional representation with zero of the number n in base k , while $\sigma_k(n)$ is the k -lexicographic representation of n .

Algorithm 8 (Conversion between Lexicographic and ZeroPositional representations). *The k -positional (with zero) and k -lexicographic representations are mutually translated, in a 1-to-1 way, by the functions $\theta_{0 \rightarrow k}$ and $\theta_{k \rightarrow 0}$ computed with the following algorithm.*

Given a string α that is a positional representation with zero in base k , and that does not start with the symbol $[0]$ (apart the string $[0]$ that is intended to have order number zero), then factorise α as a concatenation of substrings (some of α_i, γ_i can be empty strings):

$$\alpha_0 \gamma_1 \alpha_1 \gamma_2 \dots \alpha_{h-1} \gamma_h \alpha_h$$

where α factors are 0-free and γ factors have the form $[j][0]^i$ with $j > 0$ and $i > 0$. Now, let:

$$\gamma_i^* = ([j][0]^{i-1} - [1])[k]$$

where minus operation is computed in the k -positional representation with zero. Then:

$$\theta_{0 \rightarrow k}(\alpha) = \alpha_0 \gamma_1^* \alpha_1 \gamma_2^* \dots \alpha_{h-1} \gamma_h^* \alpha_h.$$

Analogously, given a string β that is a lexicographic representation in base k , factorise β as a concatenation of substrings (some of β_i, η_i can be empty strings):

$$\beta_0 \eta_1 \alpha_1 \eta_2 \dots \beta_{g-1} \eta_g \beta_g$$

where β factors are $[k]$ -free, and η factors have the form $[j][k]^i$ with $j < k$ and $i > 0$. Now, let:

$$\eta_i^o = [j+1][0]^i$$

where in the case that $j+1 = k$ the cipher $[k]$ has to be replaced by $[1][0]$. Then:

$$\theta_{k \rightarrow 0}(\beta) = \beta_0 \eta_1^o \beta_1 \eta_2^o \dots \beta_{g-1} \eta_g^o \beta_g.$$

QED.

The following proposition states the correctness of the translation functions $\theta_{0 \rightarrow k}$ and $\theta_{k \rightarrow 0}$ defined by the algorithm above.

Proposition 9.

$$\begin{aligned} \theta_{0 \rightarrow k}(\delta_k(n)) &= \sigma_k(n) \\ \theta_{k \rightarrow 0}(\sigma_k(n)) &= \delta_k(n) \end{aligned}$$

Proof. Firstly, we observe that if we order the alphabet $\{[0], [1], \dots, [k-1]\}$ with the natural order (where $[0]$ is the minimum symbol), then the number represented by a string α is the enumeration order of this string, according to the lexicographic ordering (restricted to the strings that do not start with the symbol $[0]$). Let us denote by $\omega_0(\alpha)$ the order number of α in this ordering. Then, the asserted equations are equivalent to the following ones. In other words, the translations between the two representations preserve their interpretations as numbers:

$$\begin{aligned}\omega(\theta_{0 \rightarrow k}(\delta_k(n))) &= n \\ \omega_0(\theta_{k \rightarrow 0}(\sigma_k(n))) &= n.\end{aligned}$$

In fact, the translation rules of the algorithm above guarantee that translations preserve the numerical value expressed by the strings, because they essentially express $[k]$ as $[1][0]$ and *vice versa*, but at same time, when this substitution is applied, a carry cipher $[1]$ has to be propagated (in addition or in subtraction, respectively) to the ciphers on the left of the position where substitution was performed. QED.

For example (here base 10 is denoted by X):

$$\begin{aligned}\theta_{X \rightarrow 0}(2X9X5) &= 301005. \\ \theta_{0 \rightarrow X}(301005) &= 2X9X5 \\ \theta_{X \rightarrow 0}(2XXX9XX5) &= 300010005 \\ \theta_{0 \rightarrow X}(300010005) &= 2XXXX9XX5.\end{aligned}$$

The proof of the previous proposition puts in evidence that classical positional representations can be seen as a restricted form of lexicographic representation (limited to a subset of strings over the k -base alphabet).

Corollary 10. *Given a base $k > 1$, any number has, in this base, a unique positional representation with zero.*

Proof. The unicity of lexicographic representation is obvious (see Corollary 2), therefore, the unicity of positional representation with zero follows immediately from it, via the translation $\theta_{k \rightarrow 0}$. QED.

A final remark will be useful to the discussion of the conclusive section. Even when the base of a positional system is a number greater than 10 (for example, 20 or 60), multiplication algorithms based on positional representation can be very efficient. In fact, by using some strategy of distributed carry, as the lattice (or sieve) multiplication method, originally pioneered by Islamic mathematicians, and described by Fibonacci in his *Liber Abaci*, and using only a subset of the multiplication table (and/or other tricks), even complex multiplications can be easily developed. Tables 6 and 7 give two examples, where a

$$\begin{array}{r}
427 \times \\
35 = \\
\hline
(5 \cdot 4)(5 \cdot 2)(5 \cdot 7) \\
(3 \cdot 4)(3 \cdot 2)(3 \cdot 7) \\
\hline
\begin{array}{r}
\boxed{0}''' \quad \boxed{0}'' \quad \boxed{5}' \\
(3 \cdot 2) \quad 6 \quad \boxed{(3 \cdot 5)}_5'' \\
(3 \cdot 2) \quad \boxed{1}'' \quad (3 \cdot 2) \\
\boxed{2}''' \quad \boxed{1}''' \quad \boxed{3}'
\end{array} \\
\hline
\begin{array}{r}
 5 \\
6 \\
6 1 \\
2 1 3
\end{array} \\
\hline
1
\end{array}$$

TABLE 6. A multiplication in decimal representation, developed by a method of *distributed carry*, where only multiplications by 2 and 5 are used. In this method, when a multiplication provides a carry, then it is appended to the column on the left. Multiplications by ciphers different from 2 or 5 are reduced to sums of them ($3 \cdot 7 = 3 \cdot 5 + 3 \cdot 2$). Boxes and apices are used, for clarity's sake, by decorating a carry and the multiplication generating it, with the same number of apices. The algorithm remains essentially the same for lexicographic representations (using the appropriate multiplication tables).

kind of lattice multiplication is joined to the restriction of multiplication tables (to 2 and 5 in the first case, and to 2, 5, 10 in the second case).

$[7][7] \times$		
$[35] =$		
<hr/>		
$([35] \cdot [7])([35] \cdot [7])$		
<hr/>		
$([10] \cdot [7])([10] \cdot [7])$		
$([10] \cdot [7])([10] \cdot [7])$		
$([10] \cdot [7])([10] \cdot [7])$		
$([5] \cdot [7])([5] \cdot [7])$		
<hr/>		
$[10][10]$		
$[10][10]$		
$[10][10]$		
$[35][35]$		
$[1] [1]$		
$[1] [1]$		
$[1] [1]$		
<hr/>		
$[4] [9] [5]$		

TABLE 7. The same multiplication of Table 6 realized in sexagesimal representation, with the method of *distributed carry*, and where only multiplication by 10 and 5 are used (if $[j]$ is the j -sexagesimal cipher, then $427 = [7][7]$, $35 = [35]$, $14945 = [4][9][5]$). Here the sub-notation (inside brackets) is decimal. The adoption of another representation of cyphers (for instance, that one of *Almagest*) is only matter of “syntactic sugar”.

4. CONCLUSIONS

The French word for computer is *ordinateur*. Certainly, it relates to the fundamental arithmetic ordering generated from zero by means of successor operation: $0, 1, 2, \dots$ as basis of any numerical calculus. From successor all the operations come via iteration: sums are successor iterations, multiplications are sum iterations, exponentials are multiplication iterations, differences iterate the predecessor operation (the inverse of successor ordering),

divisions iterate difference, and so on. What we discover from the previous discussion is a further intriguing connection between calculus and ordering. In fact, string orderings are intrinsically related to number positional representations, which provide efficient methods for computing the arithmetical operations.

In [6] the fundamental role of positional representation in the process of understanding numbers is explained by many specific examples and analyses. For instance, the existence of irrational numbers, a masterpiece of Greek mathematics, becomes a simple corollary of sequence representations of numbers. In fact, it is easy to prove that the division of two positive integers, in positional representation, provides a finite sequence of ciphers or an infinite sequence of ciphers that is periodic (it is a consequence of simple combinatorial arguments). Then, any rule generating an infinite sequence of ciphers that is not periodic has to denote a number that cannot be rational. More sophisticated examples involve results about real and transcendental numbers [6]. It is also worthwhile to recall here that the famous Turing's paper of 1936 [12], which was the starting point of the theory of computability, was aimed at providing a definition of *computable real number*, as algorithmically generated infinite sequence of ciphers.

The kind of positional representation that is inherent to the lexicographic notation could be the basis for explaining some intriguing aspects of ancient mathematics. In fact, we know that, even before Greek mathematics, in ancient civilisations, complex calculations were developed (Babylonians were able to manage, in base 60, numerical algorithms related to complex astronomical problems). This ability is surely impossible without an efficient number representation.

In [11] the solution is reported that was given by Fibonacci to the cubic equation $x^3 + 2x^2 + 10x = 20$ in sexagesimal notation (with an error only to the sixth fractional sexagesimal cipher (60^{-6})). This reveals an amazing mathematical mastery of computations with sexagesimal fractions, additions, multiplication, extraction of square roots, and so on. However, sexagesimal tradition goes back to the *Almagest* of Ptolemy, composed about A.D. 150, within Greek culture, where tables of chords (essentially sines) are given for every $(1/2)^\circ$ from 0° to 180° , with sexagesimal primes and seconds, which is equivalent to an approximation greater than three decimals fractions (for instance, $\sin 1 = 1'2''50'''$). Ptolemy never showed numerical computations, therefore we do not know to what extent he used positional principles in computing, but according to our previous analysis, we know that they are independent of zero. Moreover, as it is remarked in [11], Ptolemy uses symbol $^\circ$ in the first position as indication of angle degree, but in the following positions (for primes, seconds, and so on) symbol $^\circ$ is used to denote the absence of any value for that sexagesimal position, that is, what now we call zero. In conclusion, also this confirms that sexagesimal notation is a positional system, within which arithmetic efficient algorithms can be available, with the full computational power of positional notation, even without any explicit notion of zero. In passing, with Ptolemy, Greek mathematics incorporated concepts coming from oriental cultures (especially Babylonian astronomy), and, at that time, they were something well known and established. Probably, the positions of the

sexagesimal system are related to the notion of time periods, as suggested by astronomical observations (already Archimedes, investigating on the representation of big numbers, elaborated a method of number representation based on periods).

More details on the historical aspect are beyond the aims of our investigation (see [7, 5, 4] for deeper analyses), however, surely positional systems were related to the *abacus* where ciphers are expressed, in some way, by the number of some items in each compartments (compartments play the role of positions of symbols). If we consider a sort of “altimetric” variant of the abacus, we can suggest a strict relationship between positional representation with zero and lexicographic positional representation. In fact, let us encode ciphers with the height of some object, say a rod, in a vertical compartment. In this way, zero is represented by the flat position of the rod, while 9 is the highest position, at some distance from the top, which is a forbidden position (cipher positions are located at equally increasing levels). This is a “zero altimetric abacus”. If otherwise, flat position is forbidden, but at same time, the top position is the highest possible one, we pass from the zero abacus to a zeroless abacus, and all the properties that we investigated in the paper easily translate in the different, but related, ways of positioning rods in the two abaci.

In the Middle Ages, positional (decimal) notation reached maturity and a decimal mark (a point or a comma) appeared in order to extend the decimal notion to fractions. In this way, following a different track, a denotation for zero naturally could arise. In fact, if the decimal mark is a point separating the integer from the proper fractional part, then a decimal point alone (without preceding or following ciphers) is a denotation of zero.

So far we discussed some historical implications of our analysis about positional systems (with and without zero). However, this analysis is not only relevant for the past mathematics. In fact, some aspects of number representation structures and algorithms are related to emergent aspects in discrete mathematics. Firstly, the algorithm outlined in Table 6 can be easily expressed as a particular *membrane system* (or P system) in the sense of [9], where columns are membranes and the manipulation rules are transformation and movement of objects between membranes. In [8] (Chapter 2) an example is given of an abacus as a membrane system following the same idea, the multiplication method of Table 6 provides an improvement of the system in [8], because computation can fully benefit from the distributed mechanism of membrane computing. Moreover, the interest in string enumeration, as suggested by the examples of Section 1, is related to the mathematical analysis of genomes. They can be viewed as strings over the four letters A, C, G, T . When we fix a representation system of numbers, by means of sequences over the genomic alphabet, then genomes become numbers of gigantic sizes, therefore the methods of string (and number) representations have a direct impact for the investigations that aim at finding unconventional ways of considering genomes, and new perspectives to their study and understanding.

REFERENCES

- [1] Boute, R. T.: Zeroless Positional Number Representation and String Ordering. The American Mathematical Monthly, Vol 107, No. 5, 437-444, May 2000.
- [2] Chabert, J. L. et al.: A History of Algorithms. Springer, 1999.
- [3] Conway, J. H., Guy R. K.: The book of numbers. Springer-Verlag, New-York, 1996.
- [4] Ifrah, G.: The Universal History of Numbers: From Prehistory to the Invention of the Computer. Wiley, New-York, 2000.
- [5] Joseph, G. Gh. A Brief History of Zero. Iranian Journal for the History of Science, 6, 37-48, 2008.
- [6] Kaplan, R. , Kaplan E.: The art of the infinite. Penguin Books, London, 2004.
- [7] Knuth, D. E.: Seminumerical Algorithms, Vol. 2 of: The Art of Computer Programming, Addison-Wesley, New-York, 1981.
- [8] Manca, V.: Infobiotics: information in biotic systems. Springer-Verlag, Berlin-Heidelberg, 2013.
- [9] Păun, Gh.: Membrane Computing. An Introduction. Springer, 2002.
- [10] Ore, O.: Number Theory and its History. Dover Publications, Inc, New-York, 1988.
- [11] Toeplitz, O.: The Calculus: A Genetic Approach. The University of Chicago Press, 2007.
- [12] Turing, A.M.: On computable numbers with application to the entscheidungsproblem. Proc. London Math. Soc 2, 42, 230–265, 1936.